

### **AMENDMENTS TO THE CLAIMS**

This listing of claims will replace all prior versions and listings of the claims in the application:

1. (Currently Amended) A computer-implemented method for testing a software application comprising:

associating a test case class with each of a plurality of operations, wherein each operation includes a collaborative behavior of a plurality of classes and wherein the test case class is defined by an application program interface (API) framework, the API framework including a test case framework that defines valid start states and probable end states for the test case class;

receiving a hierarchically organized test scenario, the test scenario including at least one selected, nested test case class;

receiving ranking information for the test scenario, the ranking information pertaining to relative prioritization of execution of each of the selected test case classes; and

performing a test of the test scenario[-] as a function of the ranking information to determine if the test scenario is semantically correct with respect to the API framework.

2. (Cancelled)

3. (Currently Amended) The method according to claim 1, wherein the ranking information is validated to be semantically correct with respect to the API [a] framework-~~semantics~~.

4. (Original) The method according to claim 3, wherein the ranking information is validated to be semantically correct by defining valid start states and probable end states for each associated operation.

5. (Currently Amended) The method according to claim 3, wherein the ranking information is validated to be semantically correct with respect to [a] the API framework semantics by providing an editor that allows only a valid nesting of test cases.

6. (Currently Amended) A computer system for testing a software application, comprising:  
  
a storage device, the storage device storing a plurality of test case classes; and

a processor, wherein the processor is adapted to:

associate a test case class with each of a plurality of operations, wherein each operation includes a collaborative behavior of a plurality of classes and wherein the test case class is defined by an application program interface (API) framework, the API framework including a test case framework that defines valid start states and probable end states for the test case class;

receive a hierarchically organized test scenario, the test scenario including at least one selected, nested test case class;

receive ranking information for the test scenario, the ranking information pertaining to relative prioritization execution of each of the selected test case classes; and

perform a test of the test scenario as a function of the ranking information to determine if the test scenario is semantically correct with respect to the API framework.

7. (Cancelled)

8. (Currently Amended) The system according to claim 6, wherein the ranking information is validated to be semantically correct with respect to [a] the API -framework semantics.

9. (Previously Presented) The system according to claim 8, wherein the ranking information is validated to be semantically correct by defining valid start states and probable end states for each associated operation.

10. (Currently Amended) The system according to claim 8, wherein the ranking information is validated to be semantically correct with respect to [a] the API framework semantics by providing an editor that allows only a valid nesting of test cases.

11-15. (Cancelled)

16. (Currently Amended) A computer system for testing a software application comprising:  
a test module;

at least one nested test case class defined for each of a plurality of operations, wherein the operation is characterized as having a beginning and an end, wherein each operation includes a collaborative behavior of each of the classes;

a first portion for receiving first information describing valid start states and probable end states for each test case class, the valid start states and the probable end states to be defined by an application program interface (API) framework;

a second portion for receiving second information for relating at least a portion of the test case classes to reflect a particular hierarchically organized scenario for testing; and

a third portion for performing a test of the particular hierarchically organized scenario as a function of the first information and second information to determine if the scenario is semantically correct with respect to the API framework.

17. (New) A method comprising:

defining a plurality of test case classes, each test case class corresponding to an operation, each operation to include collaborative behavior of two or more development classes;

creating a test suite by organizing the test case classes hierarchically corresponding to a scenario to be tested, the test suite to include test case relationships to define the relationships between test cases;

receiving from an application program interface (API) framework information defining valid start states and probable end states for the test cases;

validating the test suite as a function of the information; and

testing the scenario to determine whether the scenario is semantically correct with respect to the API framework.